

Asymptotically Optimal Sampling-Based Algorithms for Topological Motion Planning

Jung-Su Ha, Soon-Seo Park and Han-Lim Choi

Abstract—Topological motion planning is a planning problem embedding topological concept of trajectories. In this work, we propose two asymptotically optimal sampling-based algorithms for topological motion planning: (a) a batch processing-based planner, termed Fast Marching Homology-embedded Tree star (FMHT*); and (b) an incremental anytime algorithm, termed Rapidly-exploring Random Homology-embedded Tree star (RRHT*). The methods commonly expand a graph directly in the configuration space and project the associated tree onto the *topological concept-augmented space*; the computational cost in edge computation and collision checking is significantly reduced by allowing trajectories with different topology to share the edge and collision information. Illustrative numerical examples are presented to demonstrate the validity and applicability of the proposed approach.

I. INTRODUCTION

Robotic motion planning aims to design a trajectory of robot states from a given initial state to a specified goal state through a complex configuration space. Over the last two decades, a variety of research has been conducted on the motion planning problems with various system dynamics, constraints and objectives. In a complex configuration space, trajectories can be classified according to their topological features induced by obstacles. The motion planning problem embedding topological concept of trajectories are called *topological motion planning* and they have been adopted to many applications. For example, topological inequivalent trajectories have been utilized as efficient strategy for multi-agent surveillance systems [1]–[3], as candidates of a robust trajectory for motion planning under uncertain environment [4] and as reference trajectories for stochastic optimal controller [5]. Also, a trajectory of cable attached mobile robot in specific topology class has been used to manipulate and transport multiple objects [6].

One of the most general topological concept is homology; two trajectories are in the same homology class, if the boundary formed by one trajectory together with the (opposite directional) other one does not contain any obstacles. There have been some attempts to embed the concept of homology in motion planning algorithms. Bhattacharya et al. have proposed the concept of H -signature to distinguish different homology classes of trajectories and incorporated it into a graph-search algorithm to find the optimal trajectories in various homology classes for 2–3 dimensional [7] and higher dimensional [8] configuration space; they have augmented the configuration space by H -signature and

performed A* algorithm on H -signature augmented graph. In [3,9], Pokorný et al. have proposed the algebraic topological approach to automatically distinguish different homology classes of trajectories without explicit information about obstacle positions by utilizing filtration of simplicial complexes. Also, topological task projections (TTPs) is proposed in [10] to represent topological features of high-dimensional trajectories by H -signature in 2-dimensional projected space.

The concept of H -signature [7] is valid, but it is known to be difficult for the graph-search based algorithms to handle system dynamics and high-dimensional space because of *the curse of dimensionality* from discretization of configuration space. In the motion planning literature, sampling-based algorithms have widely been studied in order to cope with such difficulties and made a lot of successes theoretically and practically [11]. Especially, Karaman and Frazzoli have proposed the incremental sampling-based algorithm, namely the Rapidly-exploring Random Tree star (RRT*) [12], and more recently, Janson et al. have proposed the Fast Marching Tree star (FMT*) algorithm [13] which utilizes batch process; both algorithms guarantee probabilistic completeness and asymptotic optimality. They have naturally extended to the planning problem with high-dimensional space and system dynamics [14]–[17]. Very few attempts, however, have been made at adopting sampling-based algorithm to topological motion planning problem whose configuration space is augmented by topological signature. Only very recent research, Winding-Augmented RRT* (WA-RRT*), extends RRT* algorithm to topological motion planning [10] by conducting an *additional* H -signature sampling step; in order to create a new node, it samples a value of H -signature as well as its configuration coordinate.

In this paper, we propose two sampling-based algorithms for optimal topological motion planning: Fast Marching Homology-embedded Tree star (FMHT*) and Rapidly-exploring Random Homology-embedded Tree star (RRHT*). They inherit the properties of the FMT* and RRT* algorithms, respectively; FMHT* is batch processing algorithm like FMT* and performs the direct dynamic programming process and lazy collision checking which dramatically accelerates the speed of the algorithm [13]; RRHT* is incremental anytime algorithm like RRT* which finds a feasible trajectory quickly by rapidly exploring the configuration space and refines the solution for allowed computation time. Both algorithm do not have *additional* sampling step; they expand a graph directly in the configuration space and project an associated tree onto the *H-signature augmented space*. As a result, the proposed algorithms share the edge information

J.-S. Ha, S.-S. Park and H.-L. Choi are with the Department of Aerospace Engineering, KAIST, Daejeon, Korea {wjdt1404, brave21i, hanlimc}@kaist.ac.kr

for every layer of H -signature space and thus significantly reduce the computational cost caused by edge computation and its collision checking which are the computational bottleneck in many cases. To demonstrate the effectiveness and applicability of the proposed algorithms, three numerical examples are presented in this work.

II. PROBLEM DESCRIPTION

This paper deals with optimal topological motion planning problem. Let \mathcal{X} and \mathcal{X}_{obs} be the configuration space and the obstacle region, respectively. The obstacle-free region is defined as $\mathcal{X}_{free} = \text{cl}(\mathcal{X} \setminus \mathcal{X}_{obs})$, where $\text{cl}(\cdot)$ denotes the closure of a given set. Let x_0 and \mathcal{X}_{goal} be the initial state and the goal region, respectively, which are inside the obstacle-free region and a continuous function $\sigma : [0, 1] \rightarrow \mathcal{X}$ be a trajectory. A trajectory is called *feasible* if it (i) starts from the initial state and reaches the goal region, i.e., $\sigma(0) = x_{init}$, $\sigma(1) \in \mathcal{X}_{goal}$ and (ii) avoids collision with obstacles, i.e., $\sigma(\tau) \in \mathcal{X}_{free}$, $\forall \tau \in [0, 1]$. Also, the cost function, $c : \Sigma \rightarrow \mathbb{R}_{\geq 0}$, is given in order to evaluate a trajectory, where Σ denotes the set of all trajectories. Then, for the given tuple $(\mathcal{X}_{free}, x_0, \mathcal{X}_{goal})$ and the cost function $c(\cdot)$, the trajectory, σ^* , is called *optimal* if the trajectory is feasible and minimizes the cost function, i.e., $\sigma^* = \text{argmin} \{c(\sigma) | \sigma \text{ is feasible}\}$ and the optimal motion planning problem is defined as to find the optimal trajectory. In addition, the problem is called *kinodynamic* motion planning problem when the resulting trajectory needs to satisfy the dynamics of the system:

$$\dot{x}(t) = f(x(t), u(t)), \quad (1)$$

where u denotes the control input defined over the control input space U .

Presence of obstacles in an environment differentiates topological classes among trajectories. When topological features of trajectories need to be considered, the problem is called *topological motion planning*. The objective of such problems can vary depending on its application. For example, the objective of the problem is to find

- 1) the set of the optimal trajectories in different topological classes;
- 2) the optimal trajectory in a specific topological class.

The resulting trajectories of former problem can be utilized as efficient strategy for multi-agent surveillance systems [1]–[3], as candidates of robust trajectories for motion planning under uncertain environment [4] or as reference trajectories for stochastic optimal controller [5] and the resulting trajectory of latter problem can be used for cable robot to manipulate and transport specified multiple objects [6].

III. TOPOLOGICAL REPRESENTATION OF TRAJECTORIES

Suppose the obstacles exist in the 2-dimensional configuration space, $\mathcal{X} \subset \mathbb{R}^2$ and σ_1 and σ_2 connect the same start and end coordinates. The two trajectories are called homologous if the boundary formed by σ_1 together with opposite directional σ_2 does not contain any obstacles.

Trajectories in different homology classes can be distinguished by using a topological signature. For this purpose, H -signature for 2-dimensional space is proposed in [7] based on Complex Analysis. Let the configuration space be represented as a subset of the complex plane \mathbb{C} , i.e., $(x, y) \in \mathcal{X} \Leftrightarrow x + iy \in \mathbb{C}$. The obstacles are also subsets of the complex plane, $\mathcal{O}_1, \mathcal{O}_2, \dots, \mathcal{O}_N \subset \mathbb{C}$, and each obstacle has one *representative point* which is denoted as $\zeta_l \in \mathcal{O}_l$, $\forall l = 1, \dots, N$. The obstacle marker function, $\mathcal{F} : \mathbb{C} \rightarrow \mathbb{C}^N$, is defined as,

$$\mathcal{F}(z) = \left[\frac{1}{z - \zeta_1}, \frac{1}{z - \zeta_2}, \dots, \frac{1}{z - \zeta_N} \right]'. \quad (2)$$

Then, H -signature, $\mathcal{H}_2 : C_1(\mathbb{C}) \rightarrow \mathbb{C}^N$, is defined as integration of a differential one-form along the trajectory:

$$\mathcal{H}_2(\sigma) = \frac{1}{2\pi} \text{Im} \left(\int_{\sigma} \mathcal{F}(z) dz \right), \quad (3)$$

where $C_1(\mathbb{C})$ is the set of all trajectories in \mathbb{C} . The H -signature is also called the winding number as it represents winding angles of a trajectory for given winding centers (obstacle representative points) [10].

If two trajectories σ_1 and σ_2 connecting the same points have the same H -signatures, i.e., $\mathcal{H}_2(\sigma_1) = \mathcal{H}_2(\sigma_2)$, they are homologous and the reverse is also true. Also, we can restrict the homology class of trajectories by defining disjoint sets of allowed and blocked H -signature, \mathcal{A} and \mathcal{B} , such that $\mathcal{U} = \mathcal{A} \cup \mathcal{B}$, where \mathcal{U} denotes the set of the H -signatures of all trajectories. By properly restricting the allowed H -signature set, the topological motion planning algorithm can secure scalability with the number of obstacles.

Especially, when the trajectory from z_1 to z_2 is *short* enough (i.e., a straight line connecting z_1 and z_2 is in same homology class), its H -signature can be calculated analytically as

$$(\mathcal{H}_2(e))_l = \frac{1}{2\pi} \text{absmin}_{k \in \mathbb{Z}} (\arg(z_2 - \zeta_l) - \arg(z_1 - \zeta_l) + 2k\pi), \quad (4)$$

where function absmin returns the value which have the minimum absolute value. The H -signature of a long trajectory also can be computed by approximating it as a piecewise linear trajectory.

The concept of H -signature is extended to 3-dimensional configuration space by exploiting the theory of electromagnetism [7] and also has been generalized to higher dimensional space [8]. On the other hand, a topological feature of a trajectory in high dimensional space can be captured in 2-dimensional projected space by using valid 2D topological task projections (TTPs) [10].

IV. SAMPLING-BASED ALGORITHMS FOR OPTIMAL TOPOLOGICAL MOTION PLANNING

A. High-level Description of Proposed Algorithms

In this section, we will propose two sampling-based algorithms for optimal topological motion planning problem: one is Fast Marching Homology-embedded Tree star (FMHT*)

and the other is Rapidly-exploring Random Homology-embedded Tree star (RRHT*). The proposed algorithms construct a graph on the configuration space and project the tree onto H -signature augmented space. They are based on Fast Marching Tree star (FMT*) and Rapidly-exploring Random Tree star (RRT*) algorithms, respectively. Both algorithms are tailored to disk-connected graphs, where for the given connection radius, two vertices are considered as neighbor, and concurrently perform graph construction and graph search; latter is key feature of sampling-based algorithm improving the scalability to a high-dimensional configuration space, because it makes the algorithms not suffer from the curse of dimensionality (the algorithms need not discretize the configuration space in advance). FMHT* is batch-type algorithm like FMT*; it generates k samples in free configuration space. Then the r -disk graph is constructed and concurrently projects the tree into H -signature augmented space in order of cost-to-arrive. With this outward moving, FMHT* performs the *direct dynamic programming recursion* with *lazy collision checking*. On the other hand, RRHT* is incremental anytime algorithm like RRT*; it incrementally samples a random vertex and creates/adds a new vertex by steering the nearest vertex in the graph. Then, RRHT* performs exhaustive-search over the graph to project a tree into H -signature augmented space.

In both algorithms, the graph in configuration space is defined by a set of vertices, V , and edges, E . Also, each vertex is composed of a state, $v.x$, and set of associated nodes $v.N$. Finally, each node $n \in v.N$ has its H -signature, $n.H$, a cost, $n.c$, and a parent node $n.parent$.

B. FMHT*: Batch Type Topology-Embedded Sampling-Based Planner

FMHT* is shown in Algorithm 1. Some required functions are described as follows: $SAMPLEFREE(k)$ function returns k random states from the free configuration space. $NEARFORWARD(V, x)$ and $NEARBACKWARD(V, x)$ functions return nearby vertices within a cost of $r_{|V|} = \gamma \left(\frac{\log |V|}{|V|} \right)^{1/d}$ (see [13]) among the set of vertices, V , from and to x , respectively, and also return corresponding optimal trajectories without considering obstacles; when the planning problem has *kinodynamic* constraints, the optimal trajectory is the solution of two point boundary value problem, which can be computed in various ways according to the system dynamics and cost [14]–[16]. $PROPAGATE(n, v)$ returns the new node, n_{new} , which is created by propagating n to the vertex v ; the new node is given as $x(n_{new}) = v$ and $n_{new}.H = n.H + \mathcal{H}(e)$, where e denotes the piece-wise straight line from $v(n)$ to v ; when H -signature of the new node is blocked (i.e., $H(n_{new}) \in \mathcal{B}$), the function does not return the new node. $OBSTACLEFREE(e)$ takes a trajectory e as an argument and checks whether it lies in obstacle free region or not. $APPENDNODE(V, N)$ adds nodes, N , to each vertex in V . $APPENDGOAL(N_{goal}, z)$ first checks if there exists a node in the goal set, N_{goal} , having the same H -signature with the new node, z , by propagating it to the *goal representation*

Algorithm 1 FMHT* algorithm

```

1:  $n.H = \mathbf{0}$ ;  $n.c = 0$ ;  $n.parent = \emptyset$ ;
2:  $v_{init}.x = x_{init}$ ;  $v_{init}.N = n$ ;
3:  $V \leftarrow \{v_{init}\} \cup SAMPLEFREE(k)$ ;  $E \leftarrow \emptyset$ ;
4:  $N_{open} \leftarrow \{v_{init}.n\}$ ;  $z \leftarrow v_{init}.n$ ;
5: while  $\sim TERMINATE(N_{goal})$  do
6:    $(V_z^f, E_{x(z), V_z^f}) \leftarrow NEARFORWARD(V, x(z))$ 
7:    $N_{temp} \leftarrow PROPAGATE(z, E_{x(z), V_z^f})$ ;
8:    $N_{near} \leftarrow N_{temp} \setminus V_z^f.N$ ;
9:    $N_{open, new} \leftarrow \emptyset$ ;
10:  for  $n \in N_{near}$  do
11:     $(V_n^b, E_{V_n^b, x(n)}) \leftarrow NEARBACKWARD(V, x(n))$ ;
12:     $Y_{temp} \leftarrow PROPAGATE(n, E_{V_n^b, x(n)})$ ;
13:     $Y_{near} \leftarrow Y_{temp} \cap V_n^b.N_{open}$ ;
14:     $y_{min} \leftarrow \underset{y \in Y_{near}}{\operatorname{argmin}} \{c(y) + Cost(e_{x(y), x(n)})\}$ 
15:    if  $OBSTACLEFREE(e_{x(n), x(y_{min})})$  then
16:       $n.c \leftarrow y_{min}.c + Cost(e_{x(y), x(n)})$ ;
17:       $n.parent \leftarrow y_{min}$ ;
18:       $N_{open, new} \leftarrow N_{open, new} \cup n$ ;
19:       $E \leftarrow E \cup e_{x(n), x(y_{min})}$ ;
20:    end if
21:  end for
22:   $N_{open} \leftarrow (N_{open} \cup N_{open, new}) \setminus \{z\}$ ;
23:   $V \leftarrow APPENDNODE(V, N_{open, new})$ ;
24:   $z \leftarrow \underset{n \in N_{open}}{\operatorname{argmin}} \{c(n)\}$ ;
25:  if  $v(z) \in X_{goal}$  then
26:     $N_{goal} \leftarrow APPENDGOAL(N_{goal}, z)$ ;
27:  end if
28: end while
29: return  $PATH(N_{goal}, T = (V, E))$ 

```

point; if not, the function appends the new node to the goal set. Finally, $TERMINATE(N_{goal})$ checks if the set of goal nodes satisfies the terminal condition of the problem; for example if the objective is to find the set of the homologous inequivalent trajectories, it checks the number of the goal nodes exceed some predefined number; if the objective is to find the trajectory in a specific topological class, it checks any goal node is in that topological class.

The algorithm operates as follows. It first creates the initial node and vertex then samples a set of states from the uniform distribution on the free configuration space, \mathcal{X}_{free} (line 1–3). Then the initial node is added to the open set and chosen as the minimum cost open node (line 4). In the main loop, the algorithm finds forward near vertices of $x(z)$ and propagates z to the forward near vertices in order to make candidates of new nodes, N_{near} (line 6–7). Then, it checks the nodes already exist in the tree and excludes the existing nodes from N_{near} (line 8). For each candidate node, the algorithm finds open backward near nodes in the tree (line 11–13) and finds the optimal connections without considering obstacles (line 14). This procedure represents *direct dynamic programming recursion* on $r_{|V|}$ -disk graph and guarantees the optimal connection of the tree in obstacle

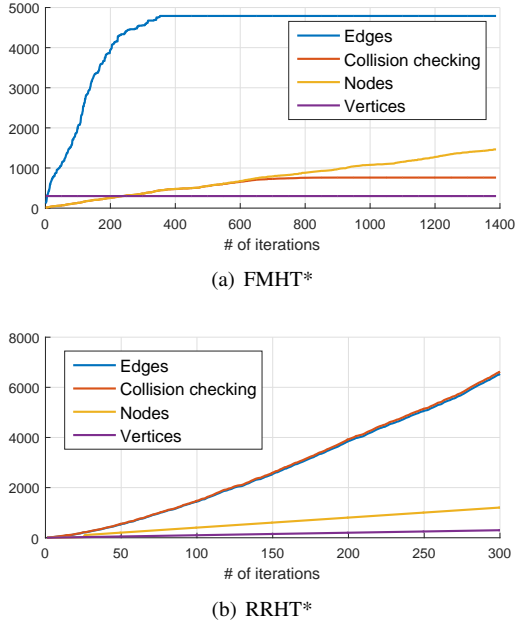


Fig. 3. The number of the edges in the graph, the collision checking, the nodes and the vertices w.r.t the number of iterations

free space from the fact that every new node must pass through a open node. Then the new node and edge are added to the tree if the connection is collision-free (line 15–19); if such connection is not collision-free, adding the new node is postponed. This *lazy collision checking* may induces sub-optimality of the tree but the number of costly collision checking is dramatically reduced; also, it is known that the cases where a suboptimal connection is made become vanishingly rare as the number of samples increases. After trying to make all connections to N_{near} , z is excluded from the open set and $N_{open,new}$ is added to the open set and to the tree (line 22–23). Then, the minimum cost open node, z , is chosen among the open set (line 24) and if z is in the goal region, the algorithm tries to append the new node to the goal set (line 25–27). The algorithm proceeds to the next iteration and when the terminal condition is satisfied, the algorithm ends and returns the tree.

C. RRHT*: Incremental Topology-Embedded Sampling-Based Planner

The RRHT* algorithm is represented in Algorithm 2. Some required functions are described as follows: `SAMPLING()` function returns a random state from the state space. `NEAREST(V, x)` returns the nearest vertex among the set of vertices, V , from x , i.e. $\text{NEAREST}(V, x) = \arg\min_{z \in V} c(e_{z \rightarrow x}^*)$, where $e_{z \rightarrow x}^*$ denotes the optimal trajectory from z to x without obstacles. `STEER(x_1, x_2)` takes two states, x_1 and x_2 as arguments and returns a new state y and a corresponding trajectory $e_{x_1 \rightarrow y}^*$ which minimizes $c(e_{y \rightarrow x_2}^*)$ while maintaining $c(e_{x_1 \rightarrow y}^*) \leq \eta$ for a prespecified one-step limitation η . `OBSTACLEFREE(e)`, `NEARFORWARD(V, x)` and `NEARBACKWARD(V, x)` functions are same as those of the FMHT* algorithm except a constant for

Algorithm 2 RRHT* algorithm

```

1:  $n.H = 0$ ;  $n.c = 0$ ;  $n.parent = \emptyset$ ;
2:  $v.x = x_{init}$ ;  $v.N = n$ ;  $(V, E) \leftarrow (\{v\}, \emptyset)$ ;
3: for  $i = 1$  to  $N_{iter}$  do
4:    $x_{rand} \leftarrow \text{SAMPLING}()$ ;
5:    $v_{nearest} \leftarrow \text{NEAREST}(V, x_{rand})$ ;
6:    $(x_{new}, e_{nearest}) \leftarrow \text{STEER}(v_{nearest}.x, x_{rand})$ ;
7:    $v_{new}.N \leftarrow \text{PROPAGATE}(e_{nearest}, v_{nearest}.N)$ ;
8:   if  $\exists v_{new}.N$  then
9:      $V \leftarrow V \cup v(x_{new})$ ;  $E \leftarrow E \cup e_{nearest}$ ;
10:     $(V_{near}^b, E_{near}^b) \leftarrow \text{NEARBACKWARD}(V, x_{new})$ ;
11:     $(V_{near}^f, E_{near}^f) \leftarrow \text{NEARFORWARD}(V, x_{new})$ ;
12:     $(V, E) \leftarrow \text{CHOOSEPARENT}(V, E, E_{near}^b)$ ;
13:     $(V, E) \leftarrow \text{REWIRE}(V, E, E_{near}^f)$ ;
14:   end if
15: end for
16: return  $\text{PATH}(N_{goal}, T = (V, E))$ 

```

Algorithm 3 CHOOSEPARENT(V, E, E_{near}^b)

```

1: for  $e_{near}^b \in E_{near}^b$  do
2:   if OBSTACLEFREE( $e_{near}^b$ ) then
3:      $E \leftarrow E \cup e_{near}^b$ ;
4:     for  $n \in v_{near}^b.N$  do
5:        $n_{new} \leftarrow \text{PROPAGATE}(e_{near}^b, n)$ ;
6:       APPENDNODE( $v(x_{new}), n_{new}$ );
7:     end for
8:   end if
9: end for
10: return  $(V, E)$ 

```

Algorithm 4 REWIRE(V, E, E_{near}^f)

```

1: for  $e_{near}^f \in E_{near}^f$  do
2:   if OBSTACLEFREE( $e_{near}^f$ ) then
3:      $E \leftarrow E \cup e_{near}^f$ ;
4:   end if
5: end for
6:  $Q \leftarrow v(x_{new}).N$ ;
7: while  $Q \neq \emptyset$  do ▷ exhaustive search
8:    $n \leftarrow \text{POP}(Q)$ ;
9:   for  $v_{near}^f$  of  $v(n)$  do
10:     $n_{new} \leftarrow \text{PROPAGATE}(e_{near}^f, n)$ ;
11:    if APPENDNODE( $v_{near}^f, n_{new}$ ) then
12:       $Q \leftarrow \text{INSERTQ}(Q, n_{new})$ ;
13:    end if
14:   end for
15: end while
16: return  $(V, E)$ 

```

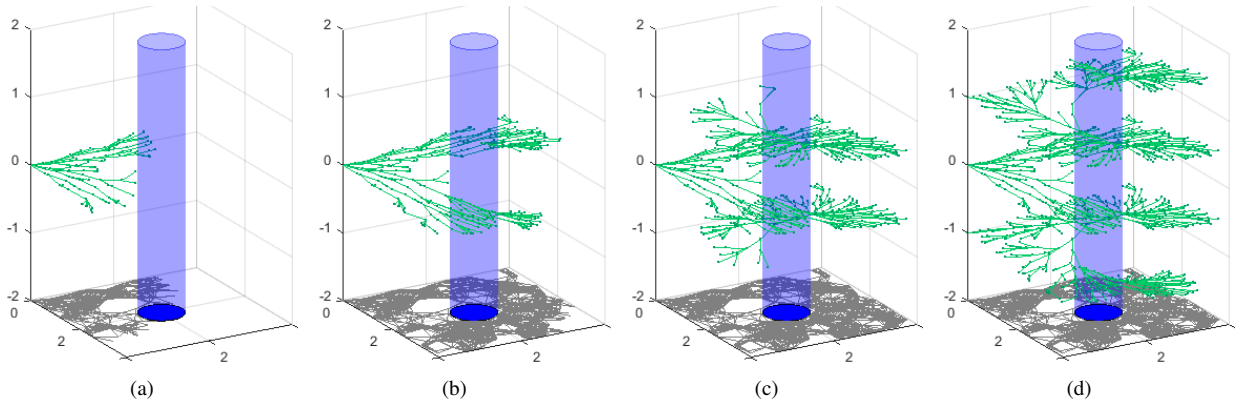


Fig. 1. FMHT* algorithm. The growth of the tree and the graph in the H -signature augmented space with (a) 100, (b) 300, (c) 600 and (d) 1000 nodes.

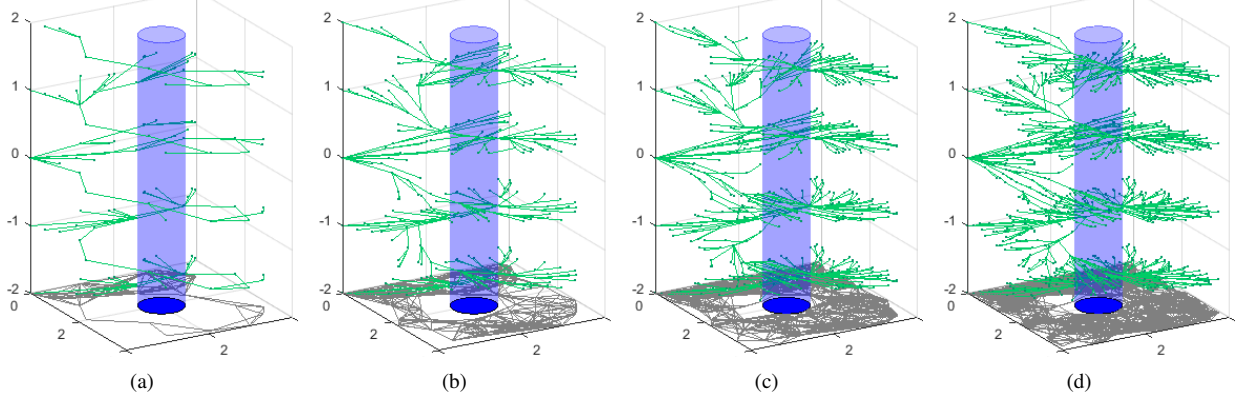


Fig. 2. RRHT* algorithm. The growth of the tree and the graph in the H -signature augmented space with (a) 108, (b) 301, (c) 601 and (d) 1001 nodes.

connection radius (see [12]). $\text{PROPAGATE}(e, n)$ and $\text{APPENDNODE}(v, n_{\text{new}})$ is similar to those of the FMHT* algorithm but they are a little different for the RRHT* algorithm. $\text{PROPAGATE}(e, n)$ returns the new node, n_{new} , which is created by propagating n through e ; the new node is given as $n_{\text{new}}.H = n.H + \mathcal{H}(e)$, $n_{\text{new}}.c = n.c + \text{Cost}(e)$ and $n_{\text{new}}.\text{parent} = n$, where $\mathcal{H}(e)$ and $\text{Cost}(e)$ denotes H -signature and cost of trajectory, e , respectively; when H -signature of the new node is blocked (i.e. $H(n_{\text{new}}) \in \mathcal{B}$), the function does not return the new node. $\text{APPENDNODE}(v, n_{\text{new}})$ imposes a partial ordering to the set of nodes in the vertex of the form:

$$n_a < n_b \Leftrightarrow (n_a.H = n_b.H) \wedge (n_a.c < n_b.c). \quad (5)$$

When above condition holds, n_a is said to be *dominated* by n_b , meaning that the paths to the root from n_a and n_b are homologous but the cost of path from n_a is smaller. This function takes a vertex, v , and new node, n_{new} as arguments and checks if the new node is dominated by any existing nodes at v . Then, if it is dominated, the function returns failure. If it is not, the function appends the new node, checks if it dominates any nodes at v and prunes when necessary. Finally, $\text{INSERTQ}(Q, n_{\text{new}})$ adds a new node into the queue and excludes a dominated node in the queue when existing.

The RRHT* algorithm operates as follows. It constructs the graph from the initial vertex, x_{init} where the initial node

is appended. In the main loop, it first samples a new state and finds the nearest vertex v_{nearest} in the graph to this sampled state (line 4–5). Then, the algorithm steers $v_{\text{nearest}}.x$ toward x_{rand} to determine x_{new} that is closest to x_{rand} and stays within a specified distance from $v_{\text{nearest}}.x$ (line 6); this procedure helps the graph to be rapidly expanded into the configuration space because the *frontier* vertex of unexplored region is more likely chosen as v_{nearest} due to Voronoi bias. The new nodes are created by propagating nodes of v_{nearest} through e_{nearest} and if such propagation creates at least one new node with allowed H -signature, the new vertex is added to the graph (line 7–9). With the successfully added new vertex, the algorithm finds/adds the new collision-free backward (and forward) edges from (and to) the graph on line 10 (and line 11) and line 1–3 in Algorithm 3 (and line 1–5 in Algorithm 4). In Algorithm 3, the algorithm attempts to create new nodes and optimally connects them to the tree (line 4–7). Also, in Algorithm 4, after adding all new edges, all nodes in the new vertex are added to the queue, Q (line 6). Then the queue is exhaustively searched using uniform cost search, like Dijkstra’s algorithm (line 7–15). These procedures make the graph project the tree into H -signature augmented space by propagating nodes in the queue and pruning with criteria in (5).

D. Discussion

It is known that because FMHT* and RRT* (which the proposed algorithms are based on) span the tree on r -disk graph which guarantees to asymptotically contain all possible trajectories through the configuration space, they have properties of probabilistic completeness and asymptotic-optimality; that is, if the planning problem is feasible, the probability of the algorithm failing to find a solution reduces to zero as the number of iterations increase, and the solution asymptotically approaches to the optimal solution as the size of samples increases. The FMHT* and RRHT* algorithms also build r -disk graph and perform graph projection into H -signature augmented space in the equivalent ways; therefore, the proposed algorithms also are probabilistically complete and asymptotically optimal.

The graph-construction/tree-projection methodology of the proposed algorithms reduces the computational cost of edge computation and its collision checking which are the computational bottleneck especially when the planning problem has kinodynamic constraints (1). The FMHT* algorithm should save the information of neighbor set and collision of the edge on line 6, 11 and 15 in Algorithm 1 and reuses such information when it is possible. This operation prevents unnecessary repeated computations and provides the important characterization of the computational complexity of the algorithm; the algorithm can *reuse/share* the edge and the collision information when it proceeds other layers of H -signature (see Fig. 1). Also, RRHT* computes optimal connections (edges) between a new node and a graph on line 10–11 in Algorithm 2 then creates/adds new nodes and rewires a tree by using the edges in Algorithm 3 and 4, respectively. As a result, like FMHT*, the edges and their collision information is *reused/shared* for all different layers of H -signature (see Fig. 2). Without these procedures, the optimal connections/collision checking should be computed/performed for all layers separately; in other words, these procedures significantly improve the scalability of the proposed algorithms for the problem having large number of obstacles (or equivalently, high-dimensional H -signature).

V. NUMERICAL EXAMPLES

In this section, we numerically investigate the computational properties and the applicability of the proposed algorithms. In the first example, a simple 2-dimensional configuration space with one obstacle is considered. Fig. 1 and 2 show how the trees are expanded into the H -augmented space by the proposed algorithms. Green lines represent the edges of the tree and dark-gray lines on the bottom denote the edge of the graph which the tree is projected by; x and y axis denote the configuration, z represents H -signature and the initial augmented state is $[0, 0, 0]^T$; in this example, H -signature is scalar because there is only one obstacle. It is shown in Fig. 1 that the tree of FMHT* is expanded in order of the cost to arrive. Also, observe that the graph is expanded only in the early phase of algorithm and the tree is projected only by the expanded graph; this implies that the algorithm do not need to compute the edges and check their collisions

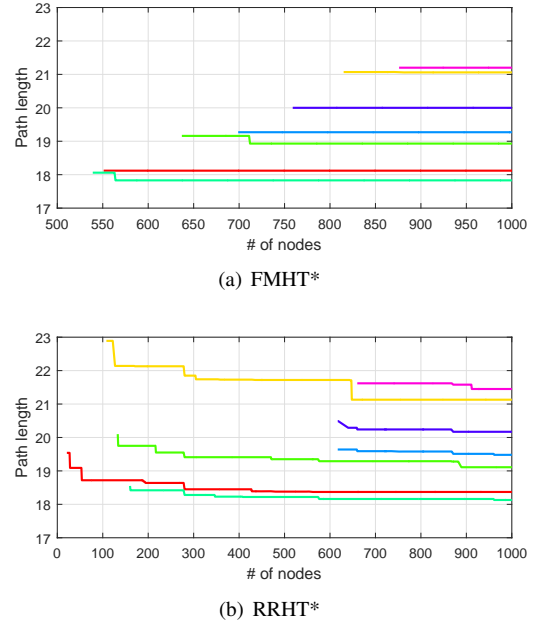


Fig. 4. The Dubin's car replanning example. Lengths of the shortest paths from $[0, 0, 0]^T$ to goal region in each homology class at each iteration.

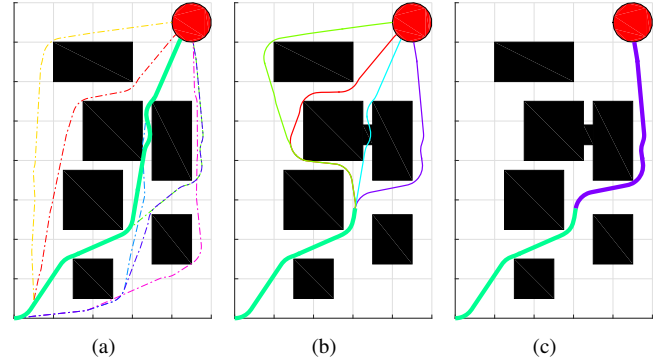


Fig. 5. The Dubin's car example. After the original trajectory found in (a) is blocked by new obstacle in (b), the robot replans its trajectory as (c).

when the tree is expanded to other H -signature layers (see Fig. 3(a)). On the other hand, Fig. 2 shows that RRHT* rapidly expands the tree to the whole space and rewires it. In addition, note that RRHT* also shares the edge information (shown as the graph) through all H -signature layers; it is shown that the graph (vertices) projects the tree (nodes) into the augmented space (see Fig. 3(b)).

The second example deals with fast replanning of Dubin's car in dynamic environment. The configuration $\mathbf{x} = [x, y, \theta]^T$ represents the position and heading of a car, which is controlled by the turn rate, u . The dynamics is given by:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} V \cos(\theta) \\ V \sin(\theta) \\ u/\rho \end{bmatrix}$$

with $V = \rho = 1$ and $|u| \leq 1$, respectively. The cost function is given as the length of the trajectory. When projecting the tree onto H -augmented space, the set of allowed H -signature

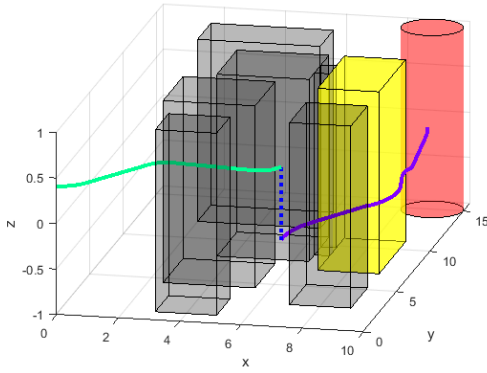


Fig. 6. Replanned trajectory of the Dubin's car in the H -signature augmented space; z axis represents the H -signature w.r.t the yellow obstacle.

value is defined as $\mathcal{A} \equiv \{z : |z_i| \leq H_{limit}, i \leq N_{obs}\}$ with $H_{limit} = 1$ to extract trajectories in physically meaningful homology classes; otherwise, infinitely many trajectories that include paths revolving around the obstacle could be obtained. In this example, the tree is expanded from the goal region, namely the *backward* tree, in order to utilize the tree as a feedback policy as in [18]. Fig. 4 shows the paths from the initial configuration $[0, 0, 0]^T$ to the goal region in each homology class until 1000 nodes are added to the trees and lengths of the paths at each iteration; it is observed that while the RRHT* algorithm finds trajectories very quickly and refines them, the FMHT* algorithm finds trajectories in order of their cost. Fig. 5 and 6 shows the replanning process in the configuration space and the resulting trajectory in the augmented space, respectively. Because a number of the topologically inequivalent trajectories were computed in lead time, the alternative trajectory can be found quickly when the originally planned trajectory become infeasible.

For the third example, we consider a two-dimensional stochastic single integrator in the environment having three obstacles. The dynamics and the cost rate are given by:

$$d\mathbf{x} = \mathbf{u}dt + b d\mathbf{w}$$

i.e. the position of a robot in the configuration space, $\mathbf{x} \in D \subset \mathbb{R}^2$, is controlled by the velocity input, $\mathbf{u} \in \mathbb{R}^2$, while the objective of control is to reach the goal region while minimizing the cost function, $J = E \left[\phi(\mathbf{x}(t_f)) + \int_0^{t_f} 1 + \mathbf{u}'\mathbf{u}dt \right]$. At the boundary (goal region or obstacles), the final cost is given as:

$$\phi(\mathbf{x}) = \begin{cases} 0 & \text{if it reaches goal region,} \\ \infty & \text{if it hits obstacles.} \end{cases} \quad (6)$$

The state is driven also by a diffusion term that contains the 2-dimensional Brownian motion; two noise levels are considered in this example for comparison:

$$b = 0.1, 0.3.$$

The path-integral approach [5,19] is applied to solve this stochastic optimal control problem; two topologically inequivalent trajectories obtained by FMHT* are used as references of path integral controller and help the controller to efficiently compute the optimal control and not to fall

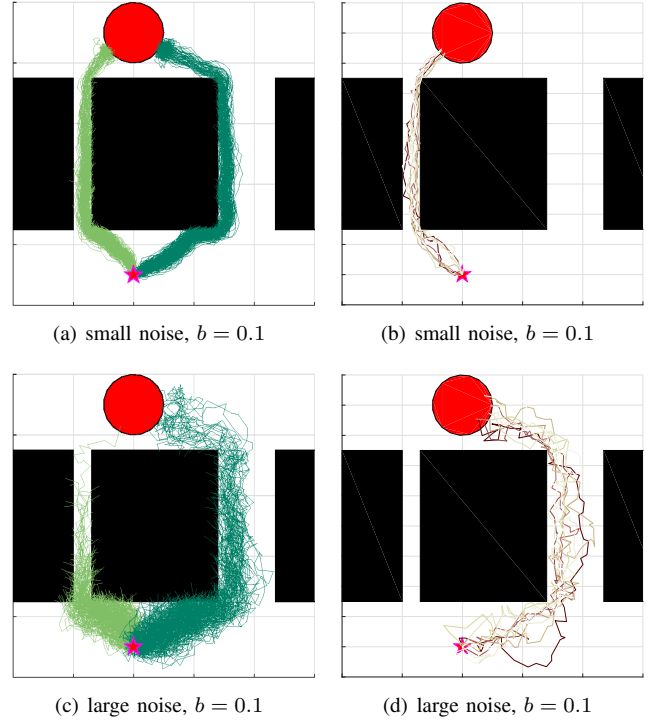


Fig. 7. The stochastic optimal control example. (a), (c) show the process of the path integral-based optimal controller and (b), (d) depict 10 resulting trajectories.

into local optima. Fig. 7 depicts the process of path integral approach and 10 resulting trajectories with two different noise level. Note that as the effect of the Brownian noise increases that the robot cannot pass through the narrow slit between the obstacles. It is observed from the figure that when the noise is not critical, the robot goes to the goal region through the narrow but nearby slit but it makes a detour as the noise increases. In summary, by considering topologically various trajectories as references, the stochastic optimal controller can lead a system to *safe and fast* path depending on the level of noise.

VI. CONCLUSIONS

In this paper, we have presented two sampling-based algorithms, FMHT* and RRHT*, for the optimal topological motion planning problem. The proposed algorithms significantly reduce the computational cost caused by edge computation and its collision checking by utilizing the graph-construction/tree-projection methodology which enables re-usage/share of edges and their collision information for all different topological trajectories. Numerical examples have demonstrated the validity of the proposed approach.

The graph-construction/tree-projection methodology has been utilized for the planning problem in *belief space* [20]. One can view this methodology as a generalization of the sampling-based algorithms; for example, FMT* and RRT* construct a graph and project a tree in the same (configuration) space while the proposed algorithms project a tree into the *topological-signature augmented space*. We expect that by using this methodology, the sampling-based planning

algorithm can be extended to solve the more general planning problems having various constraints or objectives.

ACKNOWLEDGMENT

This work was supported by Agency for Defense Development (under in part contract #UD140053JD and in part contract #UD150047JD).

REFERENCES

- [1] F. Bourgaul, A. A. Makarenko, S. B. Williams, B. Grocholsky, and H. F. Durrant-Whyte, "Information based adaptive robotic exploration," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 1, 2002, pp. 540–545.
- [2] S. Kim, S. Bhattacharya, R. Ghrist, and V. Kumar, "Topological exploration of unknown and partially known environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013, pp. 3851–3858.
- [3] F. T. Pokorny, K. Goldberg, and D. Kragic, "Topological trajectory clustering with relative persistent homology," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2016.
- [4] S. Bhattacharya, R. Ghrist, and V. Kumar, "Persistent homology for path planning in uncertain environments," *IEEE Transactions on Robotics*, vol. 31, no. 3, pp. 578–590, 2015.
- [5] J.-S. Ha and H.-L. Choi, "A topology-guided path integral approach for stochastic optimal control," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2016.
- [6] S. Bhattacharya, S. Kim, H. Heidarsson, G. S. Sukhatme, and V. Kumar, "A topological approach to using cables to separate and manipulate sets of objects," *The International Journal of Robotics Research*, 2015.
- [7] S. Bhattacharya, M. Likhachev, and V. Kumar, "Topological constraints in search-based robot path planning," *Autonomous Robots*, vol. 33, no. 3, pp. 273–290, 2012.
- [8] S. Bhattacharya, D. Lipsky, R. Ghrist, and V. Kumar, "Invariants for homology classes with application to optimal search and planning problem in robotics," *Annals of Mathematics and Artificial Intelligence*, vol. 67, no. 3-4, pp. 251–281, 2013.
- [9] F. T. Pokorny, M. Hawasly, and S. Ramamoorthy, "Topological trajectory classification with filtrations of simplicial complexes and persistent homology," *The International Journal of Robotics Research*, vol. 35, no. 1-3, pp. 204–223, 2016.
- [10] F. T. Pokorny, D. Kragic, L. E. Kavraki, and K. Goldberg, "High-dimensional winding-augmented motion planning with 2d topological task projections and persistent homology," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2016.
- [11] S. M. LaValle, *Planning algorithms*. Cambridge university press, 2006.
- [12] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.
- [13] L. Janson, E. Schmerling, A. Clark, and M. Pavone, "Fast marching tree: A fast marching sampling-based method for optimal motion planning in many dimensions," *The International Journal of Robotics Research*, pp. 883–921, 2015.
- [14] S. Karaman and E. Frazzoli, "Optimal kinodynamic motion planning using incremental sampling-based methods," in *IEEE Conference on Decision and Control (CDC)*, 2010, pp. 7681–7687.
- [15] D. J. Webb and J. van den Berg, "Kinodynamic RRT*: Asymptotically optimal motion planning for robots with linear dynamics," in *2013 IEEE International Conference on Robotics and Automation (ICRA)*, 2013, pp. 5054–5061.
- [16] J.-S. Ha, J.-J. Lee, and H.-L. Choi, "A successive approximation-based approach for optimal kinodynamic motion planning with nonlinear differential constraints," in *IEEE Conference on Decision and Control (CDC)*, 2013, pp. 3623–3628.
- [17] R. Allen and M. Pavone, "Toward a real-time framework for solving the kinodynamic motion planning problem," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 928–934.
- [18] J. Jeon, S. Karaman, and E. Frazzoli, "Optimal sampling-based feedback motion trees among obstacles for controllable linear systems with linear constraints," in *IEEE International Conference on Robotics and Automation*, 2015, pp. 4195–4201.
- [19] H. J. Kappen, "Path integrals and symmetry breaking for optimal control theory," *Journal of statistical mechanics: theory and experiment*, vol. 2005, no. 11, p. P11011, 2005.
- [20] A. Bry and N. Roy, "Rapidly-exploring random belief trees for motion planning under uncertainty," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2011, pp. 723–730.